Бордулёва Алёна*

г.Томск
НИ ТПУ. Магистр. Техническая физика

*мама в декрете :)

Geant4_Задание

1. С помощью TestEm0 определить коэффициент поглощения фотонов рентгеновской трубки (энергия 20 кэВ) в алюминии и в свинце.
2. Используя эти результаты, оценить толщину слоя алюминия и свинца, требуемых для ослабления пучка в 1000 раз.
3. Подтвердить расчетный результат с использованием моделирования по TestEm5.

Geant4_Задание 1

```
gamma (20 keV) in Aluminium (density: 2.7 g/cm3 ;   radiation length: 8.89302 cm )

 processes :                          phot                  compt                  conv                  Rayl                  total

 cross section per atom    :        139.445 barn           6.36306 barn              0 pbarn          9.33068 barn          155.139 barn


 compCrossSectionPerVolume :        8.40335 cm^-1         0.383455 cm^-1             0 cm^-1         0.562292 cm^-1        9.34909 cm^-1
 cross section per volume :         8.40335 cm^-1         0.383455 cm^-1             0 cm^-1         0.562586 cm^-1        9.34939 cm^-1
 cross section per mass   :         3.11235 cm2/g           14.202 mm2/g             0 um2/mg          20.8365 mm2/g         3.46274 cm2/g

 mean free path           :            1.19 mm             2.60787 cm       5.82593e+288 pc          1.77751 cm            1.06959 mm
       (g/cm2)            :         321.301 mg/cm2         7.04125 g/cm2    2.88022e+285 kg/cm2      4.79927 g/cm2         288.789 mg/cm2
```

```
gamma (20 keV) in Lead (density: 11.35 g/cm3 ;   radiation length: 5.61253 mm )

 processes :                          phot                  compt                  conv                  Rayl                  total

 cross section per atom    :        28784.1 barn           23.4428 barn              0 pbarn          783.137 barn          29590.7 barn


 compCrossSectionPerVolume :        949.456 cm^-1         0.773271 cm^-1             0 cm^-1         25.8321 cm^-1         976.061 cm^-1
 cross section per volume :         949.456 cm^-1         0.773271 cm^-1             0 cm^-1          25.837 cm^-1         976.066 cm^-1
 cross section per mass   :         83.6525 cm2/g          6.81296 mm2/g             0 um2/mg         2.27638 cm2/g          85.997 cm2/g

 mean free path           :         10.5324 um             1.29321 cm       5.82593e+288 pc          387.042 um            10.2452 um
       (g/cm2)            :         11.9542 mg/cm2         14.6779 g/cm2    2.88022e+285 kg/cm2      439.293 mg/cm2        11.6283 mg/cm2
```

$$I/I_0 = \exp[-(\mu/\rho)x] \ .$$

$$\mu/\rho = \sigma_{\text{tot}}/uA \ .$$

$$\sigma_{\text{tot}} = \sigma_{\text{pe}} + \sigma_{\text{coh}} + \sigma_{\text{incoh}} + \sigma_{\text{pair}} + \sigma_{\text{trip}} + \sigma_{\text{ph.n.}}$$

dJ = Jnσdx, где

J - величина потока частиц на глубине проходящего слоя;
x - толщина слоя;
n($см^{-3}$) - плотности атомов;
σ ($см^2$) - эффективное сечение взаимодействия фотонов.

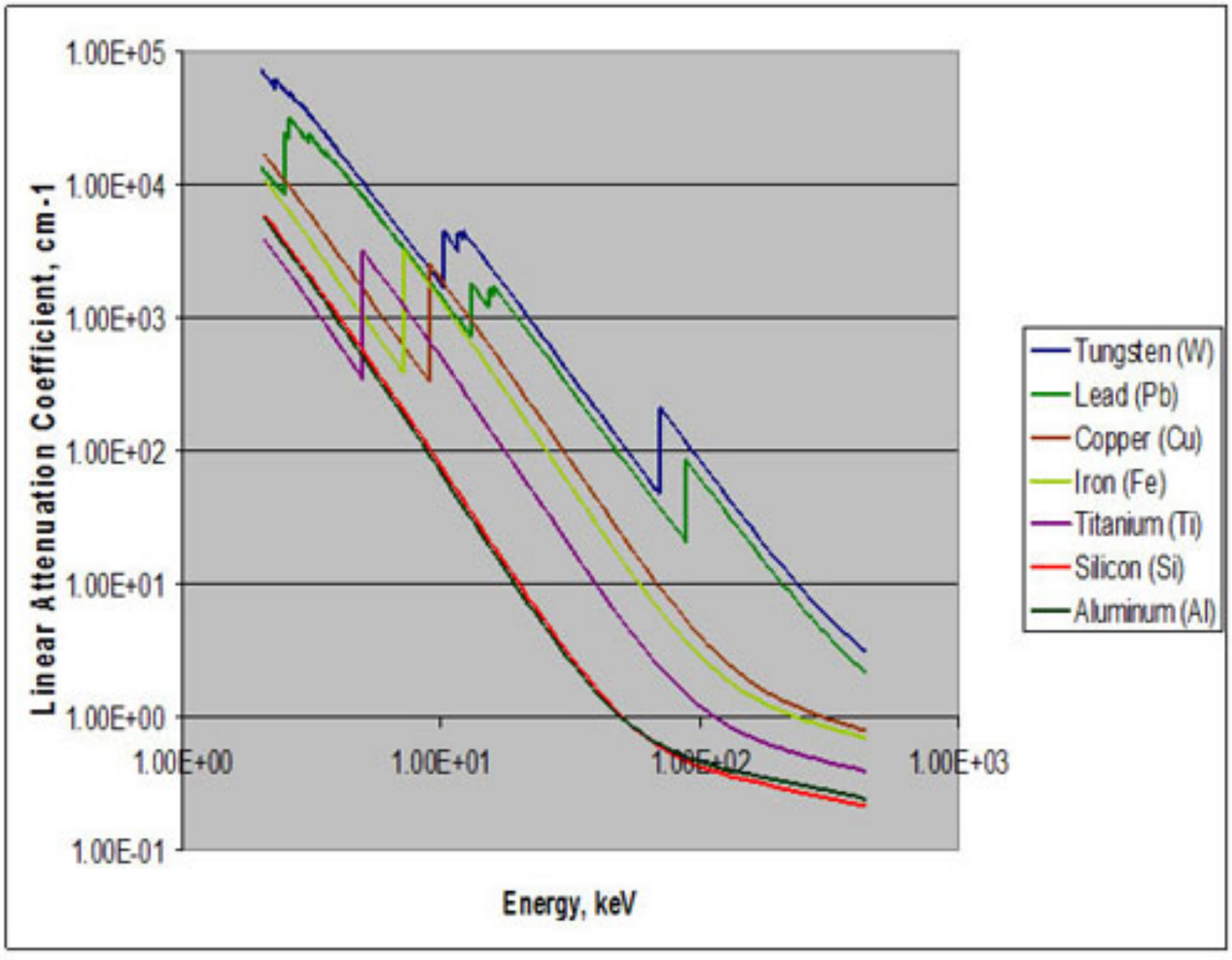
μ (Pb) = 976,14 [cm^-1]

μ (Al) = 9, 35 [cm^-1]

*http://nuclphys.sinp.msu.ru/ihem/ihem03.htm

 * https://physics.nist.gov/PhysRefData/XrayMassCoef/chap2.html

Geant4_Задание 1



* https://www.nde-ed.org/EducationResources/CommunityCollege/Radiography/Physics/attenuationCoef.htm

Geant4_Задание 2, 3

$$x\,(Pb) = -[\ln(I/I0)]/\mu = 0{,}071\ \text{мм}$$

```
The run was 50000000 gamma of 20 keV through 71 um  of Lead (density: 11.3 g/cm3 )

Total energy deposit in absorber per event = 19.94 keV +- 0.1481 eV

-----> Mean dE/dx = 2.809 MeV/cm       (0.2475 MeV*cm2/g)

From formulas :
  restricted dEdx = 0 MeV/cm   (0 MeV*cm2/g)
  full dEdx       = 0 MeV/cm   (0 MeV*cm2/g)

Leakage :  primary = 55.14 eV  +- 0.1481 eV    secondaries = 0 eV  +- 0 eV
Energy balance :  edep + eleak = 20 keV

Total track length (charged) in absorber per event = 0 fm  +- 0 fm
Total track length (neutral) in absorber per event = 10.5 um  +- 1.476 nm

Number of steps (charged) in absorber per event = 0 +- 0
Number of steps (neutral) in absorber per event = 1.028 +- 2.393e-05

Number of secondaries per event : Gammas = 0;   electrons = 0.998;   positrons = 0

Number of events with the primary particle transmitted = 0.1048 %
Number of events with at least  1 particle transmitted (same charge as primary) = 0.1048 %

Number of events with the primary particle reflected = 0.1719 %
Number of events with at least  1 particle reflected (same charge as primary) = 0.1719 %

MultipleScattering:
 rms proj angle of transmit primary particle = 0 mrad (central part only)
 computed theta0 (Highland formula)          = 0 mrad
 central part defined as +- 0 mrad;   Tail ratio = 7.256 %
```

$$x\,(Al) = -[\ln(I/I0)]/\mu = 7{,}39\ \text{мм}$$

```
======================== run summary =====================

The run was 50000000 gamma of 20 keV through 7.39 mm  of Aluminium (density: 2.7 g/cm3 )

Total energy deposit in absorber per event = 19.8 keV +- 0.272 eV

-----> Mean dE/dx = 0.0268 MeV/cm      (0.009925 MeV*cm2/g)

From formulas :
  restricted dEdx = 0 MeV/cm   (0 MeV*cm2/g)
  full dEdx       = 0 MeV/cm   (0 MeV*cm2/g)

Leakage :  primary = 195.4 eV  +- 0.272 eV    secondaries = 0 eV  +- 0 eV
Energy balance :  edep + eleak = 20 keV

Total track length (charged) in absorber per event = 0 fm  +- 0 fm
Total track length (neutral) in absorber per event = 1.173 mm  +- 164.2 nm

Number of steps (charged) in absorber per event = 0 +- 0
Number of steps (neutral) in absorber per event = 1.111 +- 4.916e-05

Number of secondaries per event : Gammas = 0;   electrons = 1.034;   positrons = 0

Number of events with the primary particle transmitted = 0.1307 %
Number of events with at least  1 particle transmitted (same charge as primary) = 0.1307 %

Number of events with the primary particle reflected = 0.8914 %
Number of events with at least  1 particle reflected (same charge as primary) = 0.8914 %
```

Higgs Boson Machine Learning Challenge

Your most recent submission

Name	Submitted	Wait time	Execution time	Score
send.csv	2 minutes to go	0 seconds	4 seconds	

Failed

```
Evaluation Exception: RankOrder values must be unique.
```


Почему?

jupyter Untitled Last Checkpoint: час назад (autosaved)

Logout

File Edit View Insert Cell Kernel Widgets Help Trusted Python 3

Run Code

In [1]:
```
import pandas as pd
```

In [2]:
```
import numpy as np
```

Считываем файл с тренировочными данными:

In [3]:
```
bh = pd.read_csv("/Users/alenaborduleva/school/Task_Jupyter/higgs-boson/training.csv", sep=",", index_col='EventId')
bh.head(6)
```

Out[3]:

EventId	DER_mass_MMC	DER_mass_transverse_met_lep	DER_mass_vis	DER_pt_h	DER_deltaeta_jet_jet	DER_mass_jet_jet	DER_prodeta_jet_jet	DER_deltar_tau
100000	138.470	51.655	97.827	27.980	0.910	124.711	2.666	3.
100001	160.937	68.768	103.235	48.146	-999.000	-999.000	-999.000	3.
100002	-999.000	162.172	125.953	35.635	-999.000	-999.000	-999.000	3.
100003	143.905	81.417	80.943	0.414	-999.000	-999.000	-999.000	3.
100004	175.864	16.915	134.805	16.405	-999.000	-999.000	-999.000	3.
100005	89.744	13.550	59.149	116.344	2.636	284.584	-0.540	1.

6 rows × 32 columns

Проверяем количество строк и столбцов:

In [4]:
```
bh.shape
```

Out[4]:
```
(250000, 32)
```

In [5]:
```
bh.columns
```

Out[5]:
```
Index(['DER_mass_MMC', 'DER_mass_transverse_met_lep', 'DER_mass_vis',
       'DER_pt_h', 'DER_deltaeta_jet_jet', 'DER_mass_jet_jet',
       'DER_prodeta_jet_jet', 'DER_deltar_tau_lep', 'DER_pt_tot', 'DER_sum_pt',
       'DER_pt_ratio_lep_tau', 'DER_met_phi_centrality',
       'DER_lep_eta_centrality', 'PRI_tau_pt', 'PRI_tau_eta', 'PRI_tau_phi',
       'PRI_lep_pt', 'PRI_lep_eta', 'PRI_lep_phi', 'PRI_met', 'PRI_met_phi',
       'PRI_met_sumet', 'PRI_jet_num', 'PRI_jet_leading_pt',
       'PRI_jet_leading_eta', 'PRI_jet_leading_phi', 'PRI_jet_subleading_pt',
       'PRI_jet_subleading_eta', 'PRI_jet_subleading_phi', 'PRI_jet_all_pt',
       'Weight', 'Label'],
      dtype='object')
```

copy selected cells

In [6]:
```
bh.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 250000 entries, 100000 to 349999
Data columns (total 32 columns):
 #   Column                      Non-Null Count   Dtype  
---  ------                      --------------   -----  
 0   DER_mass_MMC                250000 non-null  float64
 1   DER_mass_transverse_met_lep 250000 non-null  float64
 2   DER_mass_vis                250000 non-null  float64
 3   DER_pt_h                    250000 non-null  float64
 4   DER_deltaeta_jet_jet        250000 non-null  float64
 5   DER_mass_jet_jet            250000 non-null  float64
 6   DER_prodeta_jet_jet         250000 non-null  float64
 7   DER_deltar_tau_lep          250000 non-null  float64
 8   DER_pt_tot                  250000 non-null  float64
 9   DER_sum_pt                  250000 non-null  float64
 10  DER_pt_ratio_lep_tau        250000 non-null  float64
 11  DER_met_phi_centrality      250000 non-null  float64
 12  DER_lep_eta_centrality      250000 non-null  float64
 13  PRI_tau_pt                  250000 non-null  float64
 14  PRI_tau_eta                 250000 non-null  float64
 15  PRI_tau_phi                 250000 non-null  float64
 16  PRI_lep_pt                  250000 non-null  float64
 17  PRI_lep_eta                 250000 non-null  float64
 18  PRI_lep_phi                 250000 non-null  float64
 19  PRI_met                     250000 non-null  float64
 20  PRI_met_phi                 250000 non-null  float64
 21  PRI_met_sumet               250000 non-null  float64
 22  PRI_jet_num                 250000 non-null  int64  
 23  PRI_jet_leading_pt          250000 non-null  float64
 24  PRI_jet_leading_eta         250000 non-null  float64
 25  PRI_jet_leading_phi         250000 non-null  float64
 26  PRI_jet_subleading_pt       250000 non-null  float64
 27  PRI_jet_subleading_eta      250000 non-null  float64
 28  PRI_jet_subleading_phi      250000 non-null  float64
 29  PRI_jet_all_pt              250000 non-null  float64
 30  Weight                      250000 non-null  float64
 31  Label                       250000 non-null  object 
dtypes: float64(30), int64(1), object(1)
memory usage: 62.9+ MB
```

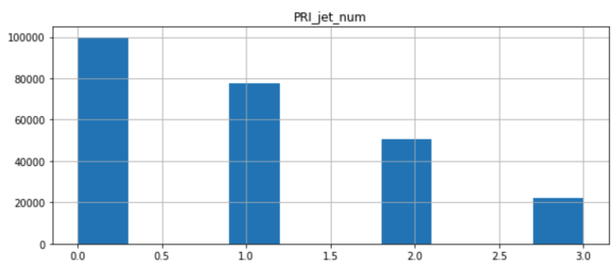
In [7]:
```
import sklearn
```

Преобразуем последний столбец в численные значения:

In [8]:
```
bh["Y"] = bh["Label"].map({ "s": 1,
"b": 0, })
```

In [9]:
```
bh[['PRI_jet_num']].hist(figsize=(10, 4))
```

Out[9]:
```
array([[<matplotlib.axes._subplots.AxesSubplot object at 0x11fca05d0>]],
      dtype=object)
```



In [10]:
```
from sklearn.ensemble import RandomForestClassifier
```

In [11]:
```
X_train = pd.DataFrame(bh)
```

In [12]:
```
X_train.drop(['Weight'], axis=1, inplace=True)
X_train.drop(['Label'], axis=1, inplace=True)
X_train.drop(['Y'], axis=1, inplace=True)
```

In [13]:
```
X_train.shape
```

Out[13]:
```
(250000, 30)
```

In [14]:
```
Y_train = (bh['Y'])
Y_train.shape
#Y_train_1D = Y_train.values
```

Out[14]:
```
(250000,)
```

In [15]:
```
weight = (bh['Weight'])
weight.head()
#weight_np = weight.to_numpy()
```

Out[15]:
```
EventId
100000    0.002653
100001    2.233584
100002    2.347389
100003    5.446378
100004    6.245333
Name: Weight, dtype: float64
```

In [16]:
```
model = RandomForestClassifier()
model.fit(X_train, Y_train, weight)
```

Out[16]:
```
RandomForestClassifier(bootstrap=True, ccp_alpha=0.0, class_weight=None,
                       criterion='gini', max_depth=None, max_features='auto',
                       max_leaf_nodes=None, max_samples=None,
                       min_impurity_decrease=0.0, min_impurity_split=None,
                       min_samples_leaf=1, min_samples_split=2,
                       min_weight_fraction_leaf=0.0, n_estimators=100,
                       n_jobs=None, oob_score=False, random_state=None,
                       verbose=0, warm_start=False)
```

In [17]:
```
task = pd.read_csv("/Users/alenaborduleva/school/Task_Jupyter/higgs-boson/test.csv", sep=",", index_col='EventId')
```

In [18]:
```
task.shape
```

Out[18]:
```
(550000, 30)
```

In [19]:
```
task.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 550000 entries, 350000 to 899999
Data columns (total 30 columns):
 #   Column                      Non-Null Count   Dtype  
---  ------                      --------------   -----  
 0   DER_mass_MMC                550000 non-null  float64
 1   DER_mass_transverse_met_lep 550000 non-null  float64
 2   DER_mass_vis                550000 non-null  float64
 3   DER_pt_h                    550000 non-null  float64
 4   DER_deltaeta_jet_jet        550000 non-null  float64
 5   DER_mass_jet_jet            550000 non-null  float64
 6   DER_prodeta_jet_jet         550000 non-null  float64
 7   DER_deltar_tau_lep          550000 non-null  float64
 8   DER_pt_tot                  550000 non-null  float64
 9   DER_sum_pt                  550000 non-null  float64
 10  DER_pt_ratio_lep_tau        550000 non-null  float64
 11  DER_met_phi_centrality      550000 non-null  float64
 12  DER_lep_eta_centrality      550000 non-null  float64
 13  PRI_tau_pt                  550000 non-null  float64
 14  PRI_tau_eta                 550000 non-null  float64
 15  PRI_tau_phi                 550000 non-null  float64
 16  PRI_lep_pt                  550000 non-null  float64
 17  PRI_lep_eta                 550000 non-null  float64
 18  PRI_lep_phi                 550000 non-null  float64
 19  PRI_met                     550000 non-null  float64
 20  PRI_met_phi                 550000 non-null  float64
 21  PRI_met_sumet               550000 non-null  float64
 22  PRI_jet_num                 550000 non-null  int64  
 23  PRI_jet_leading_pt          550000 non-null  float64
 24  PRI_jet_leading_eta         550000 non-null  float64
 25  PRI_jet_leading_phi         550000 non-null  float64
 26  PRI_jet_subleading_pt       550000 non-null  float64
 27  PRI_jet_subleading_eta      550000 non-null  float64
 28  PRI_jet_subleading_phi      550000 non-null  float64
 29  PRI_jet_all_pt              550000 non-null  float64
dtypes: float64(29), int64(1)
memory usage: 130.1 MB
```

In [20]:
```
predict_task = model.predict(task)
```

In [21]:
```
predict_task.shape
```

Out[21]:
```
(550000,)
```

In [22]:
```
print (predict_task)
```

```
[0 0 1 ... 0 0 0]
```

Всё, что представленно ниже позаимствовала из лекции Максима:

In [23]:
```
tt = model.predict_proba(task.to_numpy())
```

In [24]:
```
tt
```

Out[24]:
```
array([[1.  , 0.  ],
       [0.74, 0.26],
       [0.37, 0.63],
       ...,
       [0.99, 0.01],
       [0.8 , 0.2 ],
       [0.91, 0.09]])
```

In [25]:
```
f = open("send.csv", "w")
print ("EventId,RankOrder,Class", file=f)

rank = np.argsort(tt[:,1])

for row in range(len(task.index)):
    print("%d,%d,%s" % (task.index[row], rank[row] + 1, "s" if tt[row][1] >= tt[row][0] else "b"), file=f)
```

Спасибо за внимание